Introduction to LaTeX for AU-AUM Ph.D. Students

David A. Hughes*

May 16, 2022

1 What is LaTeX?

Quite simply, LaTeX is a program that allows you to craft beautiful documents with crisp, clear tables, figures, and mathematical formulas. Unlike other programs such as Microsoft Word, LaTeX gives its users extensive control over the look and feel of a document. Therefore, its principle benefits are that you get to make the document the way you want it to look; you get to make it as beautiful as you want it to be (which of course aids in reading and tends to make reviewers happier); and you aren't confined to the cookie-cutter documents Word tries to force you into. LaTeX has the added benefit of using a built-in bibliography, which can significantly cut down on the amount of time you spend fussing over the back-end of your document. The downsides? Well, there's a learning curve. With all that functionality, there comes a requirement that you understand a bit more about what's going on "under-the-hood." That means learning how to use the language of the software and understanding what went wrong when you made a mistake. But by the time you finish this primer in LaTeX, it's my hope that you think, like me, that the tradeoffs are more than worth it.

2 Let's Download the Software

The first thing you'll want to do is to download the appropriate software. LaTeX is free and runs on all sorts of machines, which are a couple of the things that make it an attractive bit of software. You will download a suite of software that depends on whether you own a Mac or a PC. Instructions for either follow. While LaTeX's front-end software can be loaded up on virtually any text editor, there are a few differences. The biggest one I can think of is that the text editor on a Mac has a spell-checker that PCs don't have.

*Associate Professor, *Auburn University at Montgomery*. Contact: david.hughes@aum.edu

If you own a Mac, perform the following tasks.

1. Navigate to `https://tug.org/mactex/` to acquire the appropriate software.
2. Click on the link that says "MacTeX Download."
3. Click the link called "MacTeX.pkg."
4. Follow the instructions in the dialogue box that opens.

If you own a PC, then perform these tasks.[1]

1. Navigate to `http://www.MiKTeX.org`.
2. Click on the tab that says "Download."
3. Follow the instructions from the dialogue box that opens.

Once you've acquired the relevant software, you'll be able to type into your text editor. This is the "front-end" of LaTeX, which is used to compile beautiful pdf documents. If you're using a Mac, then your text editor will likely be called TexShop. If you're using a PC, then your text editor will likely be called TexWorks.

3 Anatomy of a LaTeX Document

Open up your relevant text editor. What do you see? Not a whole lot. The blank screen and the blinking cursor can freak some folks out, but there's nothing to worry about. First things first. Let's give our file a name and save it in a place on our computers where we can easily find it.

As you think about the anatomy of your LaTeX document, here are some things to keep in mind:

1. One of the first things to know about writing in LaTeX is that you're often working with commands. In the program's language, the "backslash" is the workhorse of the command environment. Every command you type will be preceded by a backslash, and the name of the command comes right after it. For example, every time I type "LaTeX" into my text editor, I'm actually writing, `\LaTeX`, where the backslash tells the program that a command is coming, and the command itself (case sensitive) is called "LaTeX," which produces the effect you see in the document.
2. Oftentimes (though not always) commands will have subcommands or options nested within them in hard brackets or curly braces.

[1]Note: I don't own a PC, so my instructions on this part of the tutorial might not be spot on since I can't open the files up and double-check my instructions. We'll troubleshoot any errors along the way.

3. In the event that you want to write something out but you'd like LaTeX to "ignore" you, simply comment that piece out using percentage signs (examples below).

4. You're not typing a finished product but a list of demands for your machine to carry out. As such, it makes zero difference how many spaces you put between your words, for example.LaTeX is going to make those typesetting choices for you, which (quite honestly) is one of the best reasons to use the program.

5. What goes into your text editor is not a document in itself. Rather, LaTeX takes your raw text and commands and compiles a pdf document that is intended for distribution. There is a button called "Typeset" in your text editor that will accomplish this. Alternatively, you can use hotkeys to compile your document as you go. This is a good idea so that you can frequently check for any errors. On a Mac, you key in "CMD + t"; and on a PC you type "CTR + t."

6. For every document you write, you will need to save *every* file you plan to incorporate into the same (sub)folder. I'll explain this more below, but it's important to keep in mind.

Every LaTeX manuscript has a few components that make them run, and once you have these down, you're well on your way to having mastered the technology. Other types of documents might have other components (e.g., letters). But for most of what you'll use LaTeX for, these components are as follows:

1. Preamble
2. Title/Abstract
3. Body
4. Bibliography
5. Ending

Let's consider these components in turn.

3.1 Preamble

The preamble contains all of the commands and packages that allow the text of your paper to compile into a readable document. These can be packages that have been made available on the internet, or they could be commands that you have written just for whatever purpose you intended them. At its most basic level, the preamble declares what type of manuscript you are working on and calls it into existence. Oftentimes, however, preambles can be quite long. Most times, authors (myself included) simply copy and

paste the preamble from their previous documents into new ones to keep from having to type them out all over again. Unless you have commands that are in conflict with one another (and you usually don't), there's no harm in having superfluous information in your preamble.

Here's a very simple preamble, followed by some text and an ending, that will accomplish the writing of the most basic documents (try typing these commands into your text editor):

```
\documentclass[12pt]{article}

\begin{document}

Here, I am writing the most compelling piece of political science research
since Kenneth Arrow's proofs graced the literature.

\end{document}
```

The preamble to this document was contained in the command called "documentclass." It declared that we were working on an "article" with 12-point font. As an aside, LaTeXuses a type-face called "computer modern." Don't worry about overriding that default function since it's beautiful as-is, but feel free to use 11-point fonts. LaTeXunderstands several other types of document classes—some of which we'll go over during this seminar—including books, slideshows, letters, and so forth.

Generally speaking, your preamble will be quite a bit longer than just one line, however. As an example, below I have pasted the preamble I used to type the document you are presently reading:

```
\documentclass[12pt]{article}
\usepackage[affil-it]{authblk}
\usepackage{natbib}
\usepackage{hyperref}
\usepackage{verbatim}
\usepackage{setspace}
\usepackage{tabularx, calc}
\usepackage{mathtools}
\usepackage{multirow}
\usepackage{bm}
\newcounter{savefootnote}
\usepackage[left=1in,right=1in,top=1in,bottom=1in]{geometry}
\usepackage{amsmath}
\usepackage{tipa}
\usepackage{euscript}
```

```
\providecommand\phantomsection{}
\usepackage[pdftex]{graphicx}
\usepackage{blindtext}
\usepackage{tabulary}
\usepackage{wasysym}
\makeatletter
\makeatother
\usepackage{rotating}
\usepackage{longtable}
\usepackage{anyfontsize}
\usepackage{tabu}
\usepackage{hyperref}
\usepackage{epstopdf}
\usepackage{pdflscape}
\usepackage{amsthm}
\usepackage{amsmath}
\usepackage{afterpage}
\usepackage{mathrsfs}
\usepackage{amsfonts}
\usepackage{caption}
\usepackage{epsfig,graphicx}
\usepackage{anysize, setspace}
\marginsize{1in}{1in}{1in}{1in}
\setlength{\parindent}{2.75em}
\setlength{\parskip}{.5em}
\newcommand{\be}{\begin{enumerate}}
\newcommand{\ee}{\end{enumerate}}
\widowpenalty10000
\clubpenalty10000
```

I won't make use of every package and command in this document, but I've learned over the years that 99 percent of the time, these commands cover me for whatever I want to do. I'll have more to say about some of these packages and commands later on in this workshop. But let's press forward for now.

3.2 Titles and Abstracts

Once you've laid out your preamble, it's time to begin your document. Generally speaking, if you're constructing a manuscript for a seminar, conference, or journal, you'll begin with a title page and abstract. Below, I've provided some commands that reproduce the title to this document, along with some options that can make your title page a little prettier. (Note that you'll need to include the above preamble.) Note the role that curly braces play in laying out commands Also see what the tildes are doing. Note that there

are a few elements to the title page. We have the title itself, the author field, date, and abstract. I've also given you some options to take the page number off of the title page so that it looks a little more proper. Let's try to spiffy up the sample document I gave you above by using some of these commands on our own machines.

```
\begin{document}
\title{Introduction to \LaTeX~for AU-AUM Ph.D.~Students}
\author[ ]{David A.~Hughes\footnote{Associate Professor,
\emph{Auburn University at Montgomery}.
Contact: \url{david.hughes@aum.edu}} } \date{\today}
%Or you could key the date in by hand:
%\date{May 16, 2022}
\maketitle
\thispagestyle{empty} % This command is helpful if you
%want to keep a page % number off of your title page.
%If you wanted to include an abstract on your title page,
%you'd add in % something like the following:
\begin{abstract}
\noindent Your abstract would be written right here.
\end{abstract}
\newpage
\doublespacing

\end{document}
```

3.3 The Body of the Text

This is where the important stuff goes. Generally speaking, the body of your text will be broken up into sections such as an "Introduction," "Literature," "Theory and Hypotheses," "Data and Methods," and "Conclusion." LaTeX makes it easy to include section headings and subheadings—and you honestly ought to be using them given how helpful they are to your readers. Let's try to reproduce some of the body I've made below:

```
\section{The Supreme Court and the Writ of Certiorari}
%You can also write \section*{} to omit numbers from section headings.

Here is a paragraph that I am writing. When I'm done with it, I'll hit the
enter/return key twice to begin a new paragraph.

And here it is. It's the new paragraph. Latex sees the space and gets
it. It will indent the first line of the paragraph for you. And it will
break words where necessary with a dash so that you get the best-aligned
text possible. Now I want a sub-section. Here goes...
```

```
\subsection{The Role of the Solicitor General}

And this new section goes here. And so on, and so on.
```

3.4 Bibliography

Maybe one of the coolest features of LaTeX is that you can compile your bibliography in real time as you write your paper. This can be a real time-saver. To do so, we're going to use some commands both at the top (Preamble) and bottom of our paper. Additionally, we're going to have a separate file (saved in the same subfolder as our text document) that contains all the citation information. Let's give all this a bit more context.

Suppose I want to cite Kenneth Arrow's work on the impossibility theorem from 1951. That means I will have an in-text citation in addition to a citation in my references section. I might want to say, "Arrow (1951) showed that no social welfare function exists to aggregate individual preferences such that basic principles of democracy are satisfied (e.g., the absence of a dictator)."[2] To achieve this in-text and bibliographic citation, let's open up a new, blank text document (for Macs, just his "CMD + n"; for PCs, hit "CTR + n").

This will be your "BibTeX" file. Generally speaking, your entries into your bibtex file will fall into a few categories: journal articles, books, essays in books, etc. Because Arrow's work is a book, we'll write a citation for a book as such:

```
@book{arrow:1951,
  title={Social Choice and Individual Values},
  author={Arrow, Kenneth J},
  year={1951},
  publisher={Yale University Press},
  address={New Haven}
}
```

You have a few options in how you construct your bibtex entries. The first element of the command, `@book`, however, is mandatory. This tells LaTeX how to construct the citation. Then you begin a curly brace, which effectively begins the entry. I've created a label called, "arrow:1951". This part is entirely up to me. I might have just as easily said, "Arrow1951". It's just a label, and we'll make use of it back in our text file later on. Next, note that we have a few entries to fill in including a title, author, year, publisher, and place of publication. This is all pretty straightforward, but note the author construction and use of commas outside of the curly braces. Below, I give examples of a journal article and of an essay in a collected works. Be sure to save your bibtex file before you go back to your text file. Let's type in some entries into our bibtex files.

[2]Note that the in-text citation is clickable—yet another cool feature of this software.

```
@article{clarklauderdale:2010,
  title={Locating Supreme Court Opinions in Doctrine Space},
  author={Clark, Tom S$.$ and Lauderdale, Benjamin},
  journal={American Journal of Political Science},
  volume={54},
  number={2},
  pages={871--890},
  year={2010},
}

@incollection{king:1997,
 title={The Polarization of American Parties and Mistrust of Government},
 author={King, David C},
 year={1997},
 booktitle={Why People Don't Trust Government},
 editor={Nye, Joseph S$.$ and Zelikow, Philip  and King, David C$.$},
 pages={155--178},
 publisher={Harvard University Press},
 address={Cambridge}
}
```

For further documentation on the use of natbib, see `https://www.sharelatex.com/learn/Bibliography_management_with_natbib#Reference_guide`.

Okay, move back over to your text editor. In order to "call-up" references from your bibtex file, you'll need to tell LaTeX what it's looking for. The following will go in your preamble:

```
\usepackage{natbib}
```

After all your text, you'll include the following two lines of code:

```
\bibliographystyle{apsr}
\bibliography{[bibtex filename here]}
```

Okay! Now we're ready to cite stuff (remember, we have 3 citations in our bibtex file). We'll be using a package called "natbib" in order to compile our list of citations. There are few different ways to call up our citations. First, let's say I want to write the sentence: "Arrow (1951) claimed that…", then I would type:

```
\citet{arrow:1951} claimed that...
```

Note how I'm using that label I made a little while ago to tell LaTeX which entry I want. Once we've typed that in, we hit the "typeset" button. What happened? Why do we simply see a question mark where a citation ought to be? This is because LaTeX hasn't called up the citation yet. Go to the drop-down menu next to the "typeset" button and select "BibTex." Typeset that, and then move your drop-down menu back to the "LaTeX" option and typeset a couple more times. (Note that the drop-down menu differs a little for Macs and PCs.)

Below are some common ways of citing materials. For further reference, I encourage you to consult `http://merkel.texture.rocks/Latex/natbib.php`.

1. `\citep[e.g.,][]{clarklauderdale:2010}`
2. `\citet[][160]{king:1997}`
3. `\nocite{arrow:1951}`
4. `\citeyearpar{clarklauderdale:2010}`

Which produces, respectively:

1. (e.g., Clark and Lauderdale, 2010)
2. King (1997, 160)
3. [not cited but appears in references]
4. (2010)

Finally, note that as we compiled our sources how they appeared in our list of references. Keying in every citation into your bibtex file can be tiring. Thankfully, Google Scholar can make a life a lot easier. Be forewarned, however, that Google's citations are oftentimes incomplete, have errors, and are not formatted identically across resources.[3]

3.5 End of the Document

Finally, it's time to end the document. If you copied and pasted my original bit of code from above, you already had the appropriate line of code, but for the sake of exhaustiveness, here's the final line of code for your text file:

```
\end{document}
```

[3]While we won't touch on these issues today, there are also means by which you can tweak your citations to adhere to different style-guides. Quite honestly, I don't fiddle with these too much. Journals don't require submissions to conform to their style-guide until they've been accepted for publication. So you waste your time in reformatting your paper for every submission. Furthermore, editorial offices will send your manuscript to a publisher who will take your LaTeX file and put it into their own style. My point is that you shouldn't worry too much about getting the style-guide perfect for an *APSR* submission.

4 Adding Figures to your Document

LaTeX makes it quite easy to include figures, appropriately sized, with perfectly placed captions. And pictures can convey a wealth of information in a remarkably small amount of space. The general structure for a figure is as follows:

```
\centering
\begin{figure}[tb!]
\includegraphics[width=6in]{[figure name here]}
\caption{Here we provide a pithy caption for our reader.}
\label{fig:[figure label]}
\end{figure}
```

Note how we can tweak the parameters of the figure using the "width" command. Most pages are 8.5 inches wide (portrait) with one-inch margins on either side, leaving you about 6.5 inches to play with. Generally, we don't like our tables and figures to spill into the margins. If your figure is quite large, consider switching to landscape mode for it:

```
\begin{landscape}
\centering
\begin{figure}
\includegraphics[width=6in]{[figure name here]}
\caption{Here we provide a pithy caption for our reader.}
\label{fig:[figure label]}
\end{figure}
\end{landscape}
```

Let's download a picture from the internet (or use one on your computer). In order to call it up into our document, you will need to make sure that you have it in the same subfolder in which you keep your text file. I like great dane puppies, so I think I'll get a picture of one of those. Find a picture and save it (for god's sake, use an intuitive name—I'll save mine as "gdpuppies.jpg"). Keep in mind that while LaTeX can read most picture extensions, there are some, like gifs, that it cannot. Provided that you don't have more than one file of the same name, you needn't supply the file extension in your command for a figure. If you want to use a label, then we can type a reference to that in the text that allows our reader to interact with it using the command, `\ref{fig:[figure label]}`. The following code produces what's called Figure 1 above:

```
\afterpage{
\begin{figure}
\centering
\includegraphics[width=6in]{gdpuppies.jpg}
```

Figure 1: Four adorable great dane puppies who are grey

```
\caption{Four adorable great dane puppies who are grey}
\label{fig:1}
\end{figure}

}
```

Finally, getting the placement of a figure can be a little tricky at times. Generally speaking, tables and figures belong at the top of a page. Furthermore, best practices demand that a figure appear directly *above* the first time it is referenced in the text—either on that page or the one immediately preceding it—whenever possible. We can use the options in the square brackets, `\begin{figure}[tb]`, to try to drop a figure in right where we want it. Frankly, I find much of this tedious and a waste of time. Here's what I do. First, don't worry about where your figures are until you are finished writing your paper. As long as you're adding text here and there, their placement will change. Second, I rely upon a command called "afterpage." It's a really convenient means of making sure a figure or table goes right at the top of the next page (example above). Third, you can always take the traditional approach of putting all your tables and figures at the end of your manuscript. I don't like this approach because, as a reviewer, it means I flip around constantly to read your paper. But it's an acceptable form of presentation. Here's how I do that in-text: `\centerline{[Figure 4.4 about Here]}`.

Table 1: Team standings in the N.L. East

Team	Wins	Losses	Pct.	GB
Atlanta	64	51	.557	0.0
Philadelphia	65	52	.556	0.0
Washington	60	58	.508	5.5
New York	49	66	.426	15.0
Miami	48	71	.403	18.0
Source: Major League Baseball (`mlb.com`)				

[Figure 4.4 about Here]

5 Making Tables

Not to sound like a broken record here, but tables are, once again, a great reason to use LaTeX as opposed to a program like Microsoft Word. The functionality of the software allows you to craft exquisite tables. But with great flexibility comes great technicality. Tables in LaTeX are frankly a little difficult at times—but worth it. Tables are a great way to convey relationships among data to your readers that might be too cumbersome to put into a figure.

Let's begin with a very basic kind of table environment. In Table 1, I have displayed the baseball standings from the Eastern Division of the National League as of 13 August 2018 (go Braves!). Let's break this code down to see what's going on. In the first line, we call the table into existence and demand that it appear at the top of the page. On the next line, we ensure that the table shows up in the center of the page. Next comes the caption in addition to a label.[4] On the fourth line of code, we declare that we will use the "tabular" environment. There are other table environments, but this one's pretty darn functional as-is.[5] In the curly braces, we then set the number of columns, in addition to their alignment. We have one "l" followed by four "c"s, meaning we are working with 5 columns; the first is left-aligned; and the remainder are center-aligned. Our next line of code draws a horizontal line between the caption and the first row of text.

Now begins the meat of our table. We begin to fill in column headers. Each column is divided by an ampersand (&), and the end of each row is ended with *two* backslashes. After we've input the rest of our data, we draw another horizontal line. The command "multicolumn" is really useful to when we need more space. The first number in curly braces tells LaTeX how many columns we want this entry to span; the second entry tells it how that entry should be aligned; and the final entry is the text itself, which we've minimized to be the text size of a footnote. Finally, make sure every open parenthesis

[4] We call up this label just the same way as we did with figures above.

[5] Other table environments include "tabulary", "tabularx", etc. We'll discuss another type for very large tables below.

and bracket gets closed. I think it's pretty to cap off a table with two horizontal lines. End your tabular environment, and end your table. All done!

```
\begin{table}[t] % afterpage is useful among tables too to prevent page-splitting
\centering
\caption{Team standings in the N.L.~East}\label{table1}
\begin{tabular}{lcccc}
\hline
Team & Wins & Losses & Pct. & GB \\
\hline
Atlanta & 64 & 51 & .557 & 0 \\
Philadelphia  & 65 & 52 & .556 & 0 \\
Washington & 60 & 58 & .508 & 5.5  \\
New York & 49 & 66 & .426 & 15 \\
Miami & 48 & 71 & .403 & 18 \\
\hline
\multicolumn{5}{l}{\footnotesize{Source: Major League Baseball (\url{mlb.com})}}\\
\hline \hline
\end{tabular}
\end{table}
```

Right about now, you're probably thinking, "That's a lot of work for not a huge payoff." You're not wrong. That was a pretty basic table, and we could easily have accomplished the same thing (that is, convey the same information) using another environment such as Word. Where LaTeX shines is in making tables in which we need special alignments, characters, vectors of differing lengths, etc. One way to simplify making tables is to recycle old ones. When you find a format that works for you, stick with it (no use in reinventing the wheel). Ask people for their code (I honestly forget where half of my preamble came from). Or let software generate tables for you.

For example, if you use R, then you can use the package, "xtable":

```
install.packages("xtable")
library(xtable)
xtable([object name here])
```

Here's a table I made in a matter of seconds from an OLS estimation I used using data from a recent publication relating to incumbent vote-shares in state supreme court retention elections. It's the raw output, but you can see how easily you could tweak it to make it more presentable. DO NOT STOP HERE. Clean it up and make it professional looking.

```
\begin{table}[ht]
```

```
\centering
\begin{tabular}{rrrrr}
  \hline
 & Estimate & Std. Error & t value & Pr($>$$|$t$|$) \\
  \hline
(Intercept) & 73.9217 & 0.6304 & 117.27 & 0.0000 \\
  attack\_ads & -0.0034 & 0.0016 & -2.10 & 0.0371 \\
  campaign\_expenditures & 0.0000 & 0.0000 & 0.97 & 0.3324 \\
  opposition\_campaign & -10.7952 & 1.3540 & -7.97 & 0.0000 \\
  y2010 & -2.5088 & 0.9838 & -2.55 & 0.0116 \\
   \hline
\end{tabular}
\end{table}
```

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	73.9217	0.6304	117.27	0.0000
attack_ads	-0.0034	0.0016	-2.10	0.0371
campaign_expenditures	0.0000	0.0000	0.97	0.3324
opposition_campaign	-10.7952	1.3540	-7.97	0.0000
y2010	-2.5088	0.9838	-2.55	0.0116

STATA also has some canned packages to do pretty much the same thing. In your STATA console, type "findit outtex". Once you've installed the package, type "outtex" after you've run some regression. These table results tend to come out a little cleaner (they have a caption; they've spiffied up the horizontal lines; and they've thrown in some bold-face to class it up). But clearly, you'll still need to clean it up some.

```
{
\begin{table}[htbp]\centering
 \caption{Estimation results : regress
\label{tabresult regress}}
\begin{tabular}{l c c }\hline\hline
\multicolumn{1}{c}
{\textbf{Variable}}
 & {\textbf{Coefficient}}  & \textbf{(Std. Err.)} \\ \hline
attack\_ads  &  -0.003  & (0.002)\\
campaign\_expenditures  &  0.000  & (0.000)\\
opposition\_campaign  &  -10.796  & (1.354)\\
y2010  &  -2.508  & (0.984)\\
Intercept  &  73.921  & (0.630)\\
\hline
\end{tabular}
\end{table}
```

```
}
```

Table 2: Estimation results : regress

Variable	Coefficient	(Std. Err.)
attack_ads	-0.003	(0.002)
campaign_expenditures	0.000	(0.000)
opposition_campaign	-10.796	(1.354)
y2010	-2.508	(0.984)
Intercept	73.921	(0.630)

Now that we've got some of the basics down, let's drill down and make some truly beautiful tables. To begin, there's no reason ever to write numbers to more than the thousandth decimal place. Also, it really bugs me that in the tables above, the decimal place isn't lined up consistently within columns. This looks tacky. We can train LaTeX to align each column using decimal places. And who put those horizontal lines slo close to the data points? Yuck! Let's take Table 1 and implement some reforms. Note that now I'm going to use another table environment called "longtabu." The results are presented in Table 3.

```
\begin{singlespacing}
\begin{longtabu}{lcccr@{}l}
\caption{Team standings in the N.L.~East}\label{table3} \\
\hline
\noalign{\vspace{.1cm}}
Team & Wins & Losses & Pct.  & \multicolumn{2}{c}{GB} \\
\hline
\noalign{\vspace{.1cm}}
Atlanta &  64 & 51 & .557 & 0&.0 \\
Philadelphia & 65 & 52 & .556 & 0&.0  \\
Washington & 60  & 58 & .508 & 5&.5 \\
New York & 49 & 66 & .426 & 15&.0 \\
Miami & 48 & 71 & .403 & 18&.0 \\
\noalign{\vspace{.1cm}}
\hline \hline
\noalign{\vspace{.1cm}}
\multicolumn{6}{p{.45\textwidth}}{Source: Major League Baseball (\url{mlb.com})}
\end{longtabu}
\end{singlespacing}
```

Table 3: Team standings in the N.L. East

Team	Wins	Losses	Pct.	GB
Atlanta	64	51	.557	0.0
Philadelphia	65	52	.556	0.0
Washington	60	58	.508	5.5
New York	49	66	.426	15.0
Miami	48	71	.403	18.0

Source: Major League Baseball (`mlb.com`)

This looks cleaner, and the code shows us why. Let's take a look at some of the key differences. First, note the change on the command "longtabu." Before I had `{lcccc}` where now we have `{lcccr@{}l}`. What I've done is to take that last column (GB) and to make it two columns, smashed together. Look at the column for GB, and see how the code changed—an ampersand now sits right before each decimal place, which stacks the column up nicely on that decimal.[6] Next, observe how we put a small pillow of space between the horizontal lines and the data entries using the command `\noalign{\vspace{.1cm}}`. I have found that one-tenth of a centimeter is an ideal distance between these figures, but you may want to tweak the code. Next, observe how I got the multicolumn under "Source" to wrap around the width of the table. This is a really handy bit of code that you won't get with just the "multicolumn" command. Finally, note that I included (superfluous) code to single-space the table. This was unnecessary since the document is single-spaced. But when you write your papers, you should double-space, and you'll want to include this bit of code to make clean tables.

Okay, we've mastered the skill of cleaning up a simple table. Now let's take a clean, simple table and add some complexity to it. Remember, the name of the game is to present visually compelling information that is *simple* to understand. Let's build on Table 3 and show the entire National League standings, throwing some different stats in there for fun, adding some vertical line markers in there, and maybe even some symbols just for the heck of it. In Table 4, I present something slightly more complicated than before. I encourage to hack the code and to use it toward your own ends.

```
\afterpage{
\begin{singlespacing}
\begin{longtabu}{l | ccr@{}l | ccr@{}l | r@{}l}
\caption{National League team averages against each division}\label{table4} \\
  \hline
\noalign{\vspace{.1cm}}
& \multicolumn{4}{c}{Home} & \multicolumn{4}{c}{Away} &
```

[6]Note that this was unnecessary for the other columns in the table because they were already properly aligned.

```
\multicolumn{2}{c}{Total}  \\ \cline{2-11}
\noalign{\vspace{.1cm}}
Team & Wins & Losses & \multicolumn{2}{l}{$\Delta$G}  &  Wins & Losses &
\multicolumn{2}{l}{$\Delta$G} & \multicolumn{2}{c}{WCGB} \\
\noalign{\vspace{0.1cm}}
\hline
\noalign{\vspace{0.1cm}}
\bf{\emph{NL East}} \\
\noalign{\vspace{0.1cm}}
Atlanta & 30 & 24 & +&6 &  34 & 27 & +&6 & 0&.0 \\
Philadelphia & 38 & 18 & +&10 & 27 & 34 & $-$&7 & 0&.0  \\
Washington & 30 & 28 & +&2 & 30 & 30 & &0 &  5&.5 \\
New York & 24 & 37 & $-$&13 & 25 & 29 & $-$&4 & 15&.0  \\
Miami &  28 & 35 & $-$&7 & 20 & 36 & $-$&16 & 18&.0  \\
\bf{\emph{NL Central}} \\
\noalign{\vspace{0.1cm}}
Chicago &  37 & 22 & +&15 & 31 & 27 & +&4 & 0&.0  \\
Milwaukee & 36 & 24 & +&12 & 31 & 30 & +&1 & 0&.0 \\
St.~Louis & 29 & 26 & +&3 & 34 & 29 & +&5 & 2&.5  \\
Pittsburgh & 33 & 29 & +&4 & 28 & 29 & $-$&1 & 5&.0   \\
Cincinatti & 28 & 32 & $-$&4 & 24 & 34 & $-$&10 &  13&.5  \\
\bf{\emph{NL West}} \\
\noalign{\vspace{0.1cm}}
Arizona  & 32 & 29 & +&3 & 33 & 25 & +&8 & 0&.0  \\
Los Angelos  & 31 & 28 & +&3 & 33& 27 & +&4 & 2&.0  \\
Colorado & 31 & 27 & +&4 & 32 & 28 & +&4 & 2&.5   \\
San Francisco & 34 & 26 & +&8 & 25 & 34 & $-$&9 & 7&.0   \\
San Diego & 22 & 37 & $-$&15 & 26 & 35 & $-$&9 & 18&.5  \\
\noalign{\vspace{0.1cm}}
\hline
\hline
\noalign{\vspace{0.1cm}}
\multicolumn{10}{p{.71\textwidth}}{Notes: All data are drawn from Major
League Baseball (\url{mlb.com}); $\Delta$G denotes difference between
games won and los; WCGB denotes how many games out of first a
team is for a berth in the wild card game.}
\end{longtabu}
\end{singlespacing}
}
```

The last thing I'd like to note in this section is that the "longtabu" environment is really helpful if you have a table that absolutely must span more than one page. Ordinarily, this is frowned upon, but I can attest to it's usefulness. If you can avoid this

Table 4: National League team averages against each division

	Home			Away			Total
Team	Wins	Losses	ΔG	Wins	Losses	ΔG	WCGB
NL East							
Atlanta	30	24	+6	34	27	+6	0.0
Philadelphia	38	18	+10	27	34	−7	0.0
Washington	30	28	+2	30	30	0	5.5
New York	24	37	−13	25	29	−4	15.0
Miami	28	35	−7	20	36	−16	18.0
NL Central							
Chicago	37	22	+15	31	27	+4	0.0
Milwaukee	36	24	+12	31	30	+1	0.0
St. Louis	29	26	+3	34	29	+5	2.5
Pittsburgh	33	29	+4	28	29	−1	5.0
Cincinatti	28	32	−4	24	34	−10	13.5
NL West							
Arizona	32	29	+3	33	25	+8	0.0
Los Angelos	31	28	+3	33	27	+4	2.0
Colorado	31	27	+4	32	28	+4	2.5
San Francisco	34	26	+8	25	34	−9	7.0
San Diego	22	37	−15	26	35	−9	18.5

Notes: All data are drawn from Major League Baseball (`mlb.com`); ΔG denotes difference between games won and los; WCGB denotes how many games out of first a team is for a berth in the wild card game.

problem by tinkering with text size (still keeping it readable), etc., I'd at least try that route first.

6 Math Mode

One of the primary reasons that I use LaTeX is because I delve in statistics, along with applied game theory. Therefore, writing mathematical formulas is important to me, and it's doubly important that my readers know what I'm trying to convey. Writing clean, crisp formulas goes a long ways toward achieving those ends.

6.1 Basics of Math Mode

Generally speaking, there are two ways to go into "math mode." The first is an in-text method. This means that we keep the math within the sentence in which it appears. For example, I might want to simply point out that $\sqrt{9} = 3$. Here, we can simply flank our mathematical turn-of-phrase by dollar-signs: `$\sqrt{9}=3$`. The second way we can enter into math-mode is to call up an equation. There's more than one way to this, but one of the simplest results in the following:

$$\sigma = \frac{\sum_i^n \big(x_i - \bar{x}\big)^2}{n-1} \tag{1}$$

Equation 1 looks nice (by the way, we label those the same way as tables and figures), so let's take a look at how that works:

```
\begin{equation}\label{equ1}
\sigma = \frac{\sum_i^n\bigr(x_i - \bar{x}\bigr)^2}{n-1}
\end{equation}
```

You can see how we called our equation into existence there with the "begin" and "end" functions. Inside that call, we simply laid out the equation we wanted (more on these commands below). Finally, we can use another function for writing equations which is useful when it comes to stacking equations up on top of on another. The example below also shows you how to suppress an equation number:

$$\begin{aligned} Pr(Y_i = 1 \mid X_1) &= Pr(\epsilon_i \leq \beta_0 + \beta_1 X_{1i}), \\ Pr(Y_i = 1 \mid X_1) &= \Phi(\beta_0 + \beta_1 X_{1i}). \end{aligned}$$

6.2 Mathematical Symbols and Operators

It would probably be silly of me to try to go over every symbol and operator you might need as you chart your course through LaTeX. Luckily, others have blazed a trail for us. Here's a really handy cheat-sheet for all manner of symbols and operators: `https://reu.dimacs.rutgers.edu/Symbols.pdf`. Also (and this is a lot of fun), you should go draw the symbol you want at `http://detexify.kirelabs.org/classify.html`. Oftentimes, the command is exactly what you'd think it would be. For example, you get a lower case ω with `\omega` and an upper-case Ω with `\Omega`. One bit of code that might come in handy (and that you might be less likely to run into in your frantic google searches, is

how to compose matrixes. For example, suppose we wanted to multiple a 4×4 identity matrix by a 4×1 vector of ones:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}, \tag{2}$$

which is produced by the following lines of code:

```
\begin{equation}
\begin{bmatrix}
1 & 0 & 0 & 0\\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix} \times
\begin{bmatrix}
1\\
1\\
1\\
1
\end{bmatrix} =
\begin{bmatrix}
1\\
1\\
1\\
1
\end{bmatrix}
\end{equation}
```

Of course, we're really only scratching the surface here with all the possibilities of LaTeX's math-mode. One final command that you may find useful, especially if you adopt formal modeling as a pastime, is the "proof" environment:[7]

Proof. To prove the proposition, it is sufficient to note that $\frac{\partial f(x)}{y} > 0$. □

We get that bit of loveliness using the commands:

```
\begin{proof}
To prove the proposition, it is sufficient to note that
$\frac{\partial f(x)}{y} > 0$.
\end{proof}
```

[7]There are other proof environments, but I've never once found a bug in this one.

7 LaTeX Environments

Lists can be a great way to enumerate your thoughts—even in a paper you plan to submit for review somewhere. LaTeX has a few key environments. The first is simply a set of bullet points:

```
\begin{itemize}
\item{Hypothesis 1: $X \rightarrow Y$.}
\item{Hypothesis 2: $X \leftarrow Y$.}
\item{Hypothesis 3: $\frac{\partial f(x)}{y} < 0$.}
\end{itemize}
```

- Hypothesis 1: $X \rightarrow Y$.
- Hypothesis 2: $X \leftarrow Y$.
- Hypothesis 3: $\frac{\partial f(x)}{y} < 0$.

The second environment of interested is an enumerated list. It's put together very similarly as the previous one:

```
\begin{enumerate}
\item{Onions}
\item{Peppers}
\item{Celery}
\end{enumerate}
```

1. Onions
2. Peppers
3. Celery

Finally, we also have what are called descriptive lists. For example, we might want to create a list of every state capital:

Alabama: Montgomery

Alaska: Anchorage

Arizona: Phoenix

Arkansas: Little Rock

It's worth noting that you can also nest your lists inside one another.

```
\begin{itemize}
\item{Item (a)}
\begin{itemize}
\item{Item (a)1}
\end{itemize}
\end{itemize}
```

- Item (a)
  - Item (a)1

8 Creating Your Own LaTeX Commands

Let's face it: all these typing of commands can get repetitive (especially if you make a lot of lists). One way to avoid typing the same commands over and over is to use the "newcommand" function in your preamble. Suppose you wanted to make a shortcut for an enumerated list:

```
\newcommand{\be}{\begin{enumerate}}
\newcommand{\ee}{\end{enumerate}}
```

Now, I can simply use this new command to quickly write a list:

```
\be
\item{Item 1}
\item{Item 2}
\item{Item 3}
\ee
```

1. Item 1
2. Item 2
3. Item 3

The same logic could apply to virtually any command so long as you don't have multiple definitions of the same command names.

The "newcommand" function is designed to let you make *new* commands. But suppose you wanted to hijack the definition of a command that already exists? That's where "renewcommand" comes in. It allows you to overwrite command definitions. Don't do this unless you are very sure of what you're doing.

9 Some Useful Writing Tips and Commands

After years of using LaTeX, there are a number of insights I've had that I think any novice of the software ought to know. As such, this section represents more of a grab bag of selected wisdom than anything else.

1. Here are a few commands that can help you get the text style you're looking for:
   - Italics: `\emph{text you want italicized}`
   - bold-face: `\bf{text you want bold-faced}`
   - underline: `\underline{text you want underlined}`
   - underline with nothing over it: `$\rule{1cm}{0.15mm}$`
2. Footnotes
   - `I wrote a sentence.\footnote{Oh no, I needed to clarify something.}`
3. Quotations
   - This is one of the most frequent mistakes I see in other people's LaTeXfiles. When you open a quote, you don't hit your quote key (if you do, it'll be turned around backwards). Rather, open a quote by hitting the accent key (to the left of the 1 on your keyboard with no shift applied) twice. To close a quote, then go to your quote key. ``Quote this."
   - There's also a command for block quotes, which is simply "quote." This is handy since you don't have to worry about adjusting margins sizes. I do, however, single-space my block quotes.

     ```
     \begin{quote}
     Here's a lengthy quote of questionable use to the overall point I'm
     allegedly making.
     \end{quote}
     ```
4. How to change around paragraphs and pages:
   - If you really need to start a new page, then `\newpage` will end a page where you type it, and what you type next will be on a new page.
   - Sometimes, you want to move your cursor down a few lines, but you don't want to start a new paragraph, per se. Then you can use the command `\noindent` to take away the automatic indentation.
   - LaTeX also allows you to change the style of the page you're on. You can make use of the command, `\marginsize{1in}{1in}{1in}{1in}`, to tweak the margins.

- You can also twiddle with the amount of indentation you want and the space you have between paragraphs:

  ```
  \setlength{\parindent}{3em}
  \setlength{\parskip}{1em}
  ```

5. Periods

   - Yes, periods. This probably is the most common bug I see among seasoned TeXers. When LaTeX sees a period, it assumes it reached the end of a sentence. And being the old soul that it is, it adds an extra space before the next word. Now, 99% of the time, you put a period in to denote the end of a sentence, so that's fine. But if you don't want that extra space, then use a tilde: `Aug.~13, 2018`.

6. Symbols LaTeX thinks are commands but that you sometimes want to use.

   - Suppose you really wanted to write $75. If you just write that, LaTeXis going to think that you began a math phrase, and it will give you an error for not having closed the expression. The same applies for percentage signs, ampersands, etc. To get around these issues, use the backslash: `\$75`. Now, the exception to this rule is the backslash itself. Since the symbol is a command for everything, printing one (\) is counterintuitive, `\textbackslash`.

7. Verbatim Text

   - Suppose, as today, I want to type out the commands I'm showing you without having LaTeXgo ahead and print the end product. That's where the verbatim environment. There are two ways to do this. The first is in-text. Here you can casually drop some code into a sentence. To do so, use the command, \verb!and everything between the exclamation points are verbatim!. The other way to get verbatim text is to declare a verbatim environment:
     \begin{verbatim}

     ```
     Verbatim text, $ & %, goes here
     ```

     \end{verbatim}

8. Hyperlinks

   - I link to websites all the time these days, especially if I want to drop a footnote, for example, and point a reader toward the website from which I collected my data. LaTeX allows you to easily create clickable links: `\url{[web address]`.

10 Conclusion

Hopefully, this primer has taught you a few key principles about writing article-class documents in LaTeX. Once you've mastered its elements, I think you'll agree that it creates a much more elegant manuscript than do other software. Perhaps more to the point, it is dang near infinitely flexible to whatever you're trying to accomplish. In this overview, I've discussed the issues I deal with the vast majority of the time when I'm writing papers. But you'll run into errors, sure enough. Quite honestly, Google searches help. But so do the error messages. Reward yourself for cracking a hard bit of code. Save everything. Put it someplace you can find it again, and assign files names that are relatively easy for you to remember.

References

Arrow, Kenneth J. 1951. *Social Choice and Individual Values*. New Haven: Yale University Press.

Clark, Tom S. and Benjamin Lauderdale. 2010. "Locating Supreme Court Opinions in Doctrine Space." *American Journal of Political Science* 54(2):871–890.

King, David C. 1997. The Polarization of American Parties and Mistrust of Government. In *Why People Don't Trust Government*, ed. Joseph S. Nye, Philip Zelikow and David C. King. Cambridge: Harvard University Press pp. 155–178.